

# EXAMUNION

Certified IT Exam Material Authority



Accurate study guides, High passing rate!  
We offer free update service for one year!

<http://www.examunion.com>

**Exam** : **CKA**

**Title** : Certified Kubernetes  
Administrator

**Version** : DEMO

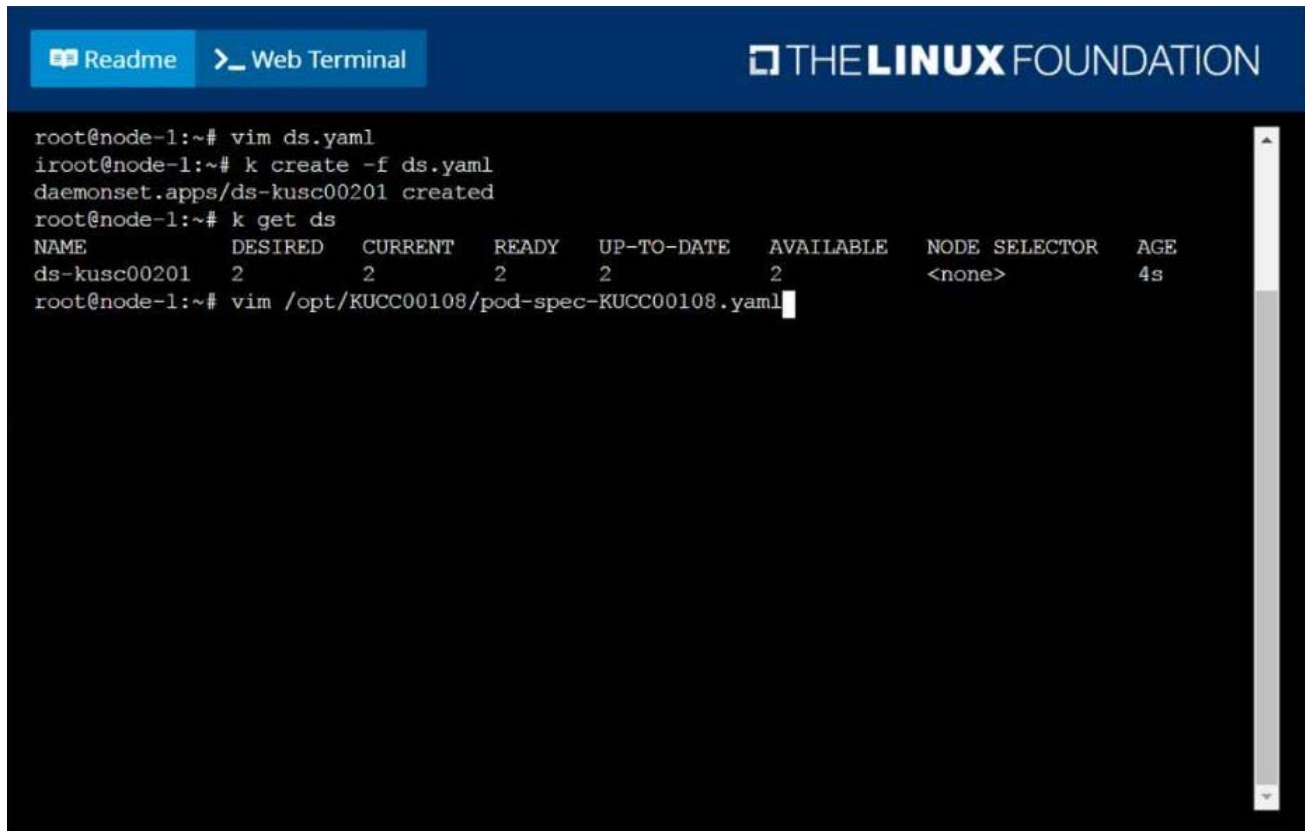
## 1.CORRECT TEXT

Perform the following tasks:

- ⇒ Add an init container to hungry-bear (which has been defined in spec file /opt/KUCC00108/pod-spec-KUCC00108.yaml)
- ⇒ The init container should create an empty file named/workdir/calm.txt
- ⇒ If /workdir/calm.txt is not detected, the pod should exit
- ⇒ Once the spec file has been updated with the init container definition, the pod should be created

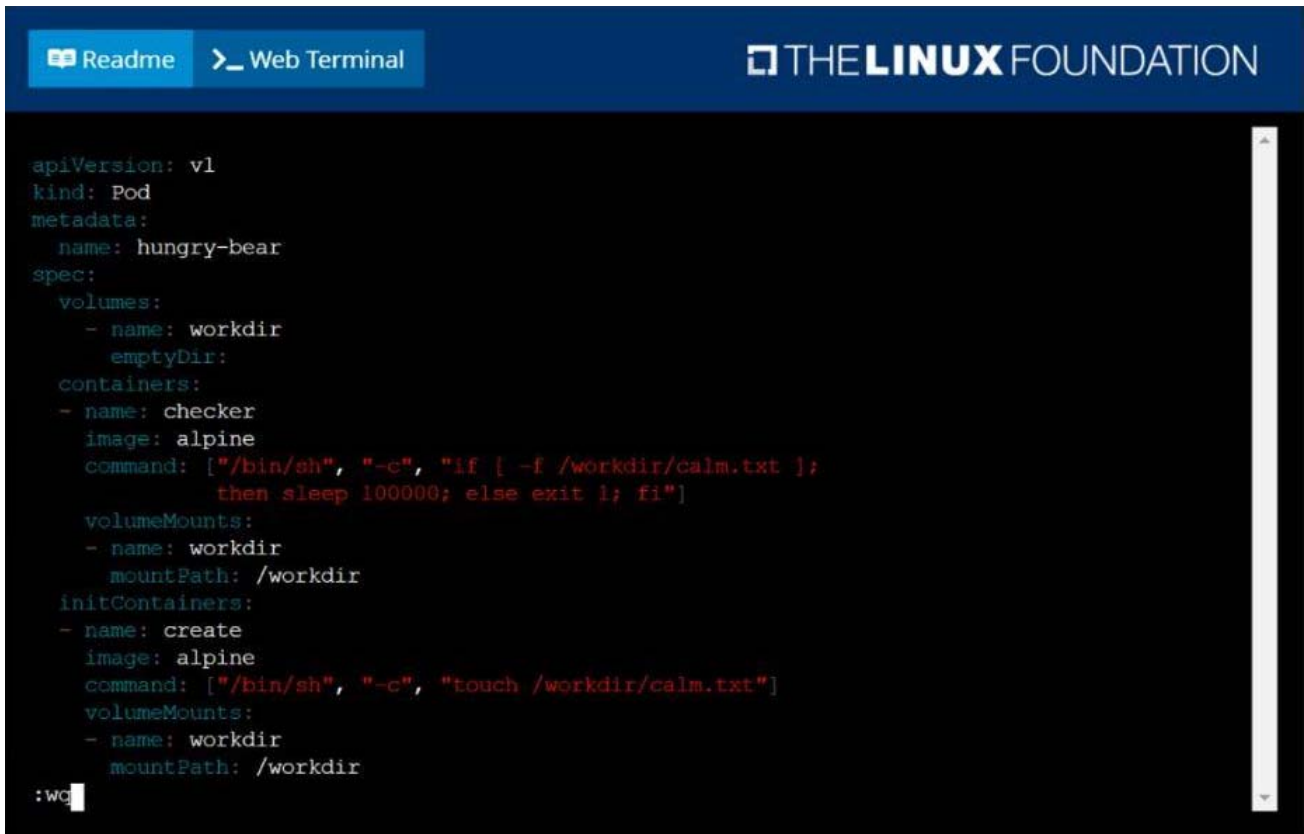
**Answer:**

solution



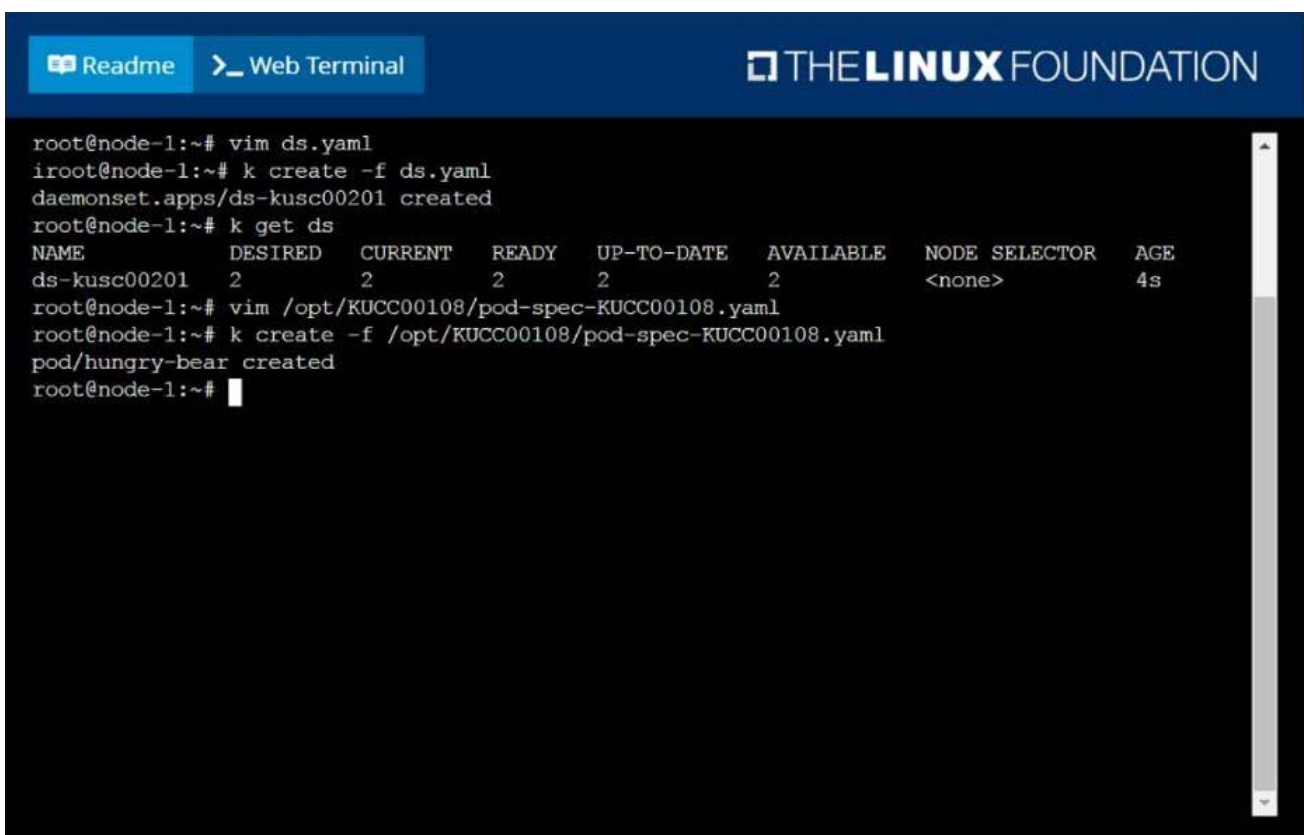
The screenshot shows a web terminal interface with a dark background. At the top, there are two buttons: 'Readme' and 'Web Terminal'. The terminal content is as follows:

```
root@node-1:~# vim ds.yaml
iroot@node-1:~# k create -f ds.yaml
daemonset.apps/ds-kusc00201 created
root@node-1:~# k get ds
NAME           DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE SELECTOR  AGE
ds-kusc00201   2        2        2      2           2          <none>         4s
root@node-1:~# vim /opt/KUCC00108/pod-spec-KUCC00108.yaml
```



The screenshot shows a web terminal interface with a dark background. At the top, there are two buttons: 'Readme' and 'Web Terminal'. To the right, the 'THE LINUX FOUNDATION' logo is displayed. The terminal content shows a YAML configuration for a pod named 'hungry-bear'. The configuration includes a 'checker' container with a command to check for a file and sleep, and an 'initContainers' section with a 'create' container that touches a file. A cursor is visible at the end of the last line.

```
apiVersion: v1
kind: Pod
metadata:
  name: hungry-bear
spec:
  volumes:
  - name: workdir
    emptyDir:
containers:
- name: checker
  image: alpine
  command: ["/bin/sh", "-c", "if [ -f /workdir/calm.txt ];
            then sleep 100000; else exit 1; fi"]
  volumeMounts:
  - name: workdir
    mountPath: /workdir
initContainers:
- name: create
  image: alpine
  command: ["/bin/sh", "-c", "touch /workdir/calm.txt"]
  volumeMounts:
  - name: workdir
    mountPath: /workdir
:WC
```



The screenshot shows a web terminal interface with a dark background. At the top, there are two buttons: 'Readme' and 'Web Terminal'. To the right, the 'THE LINUX FOUNDATION' logo is displayed. The terminal content shows a series of commands and their outputs, including creating a daemonset, listing it, and creating a pod. A table is shown for the daemonset listing.

```
root@node-1:~# vim ds.yaml
iroot@node-1:~# k create -f ds.yaml
daemonset.apps/ds-kusc00201 created
root@node-1:~# k get ds
NAME          DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE SELECTOR  AGE
ds-kusc00201  2        2        2      2          2          <none>         4s
root@node-1:~# vim /opt/KUCC00108/pod-spec-KUCC00108.yaml
root@node-1:~# k create -f /opt/KUCC00108/pod-spec-KUCC00108.yaml
pod/hungry-bear created
root@node-1:~#
```

2.CORRECT TEXT

Create a deployment spec file that will:

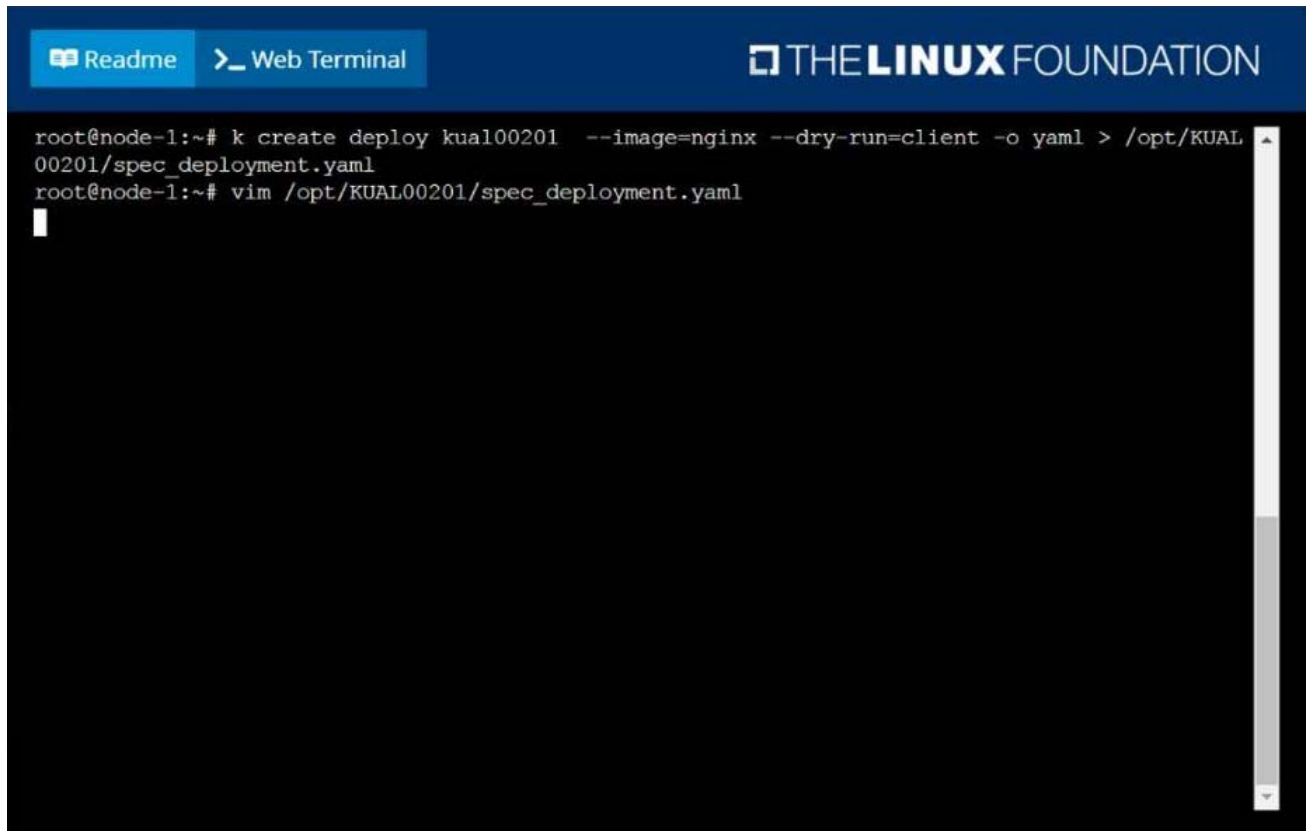
- ☞ Launch 7 replicas of the nginx Image with the labelapp\_runtime\_stage=dev
- ☞ deployment name: kual00201

Save a copy of this spec file to /opt/KUAL00201/spec\_deployment.yaml  
(or /opt/KUAL00201/spec\_deployment.json).

When you are done, clean up (delete) any new Kubernetes API object that you produced during this task.

**Answer:**

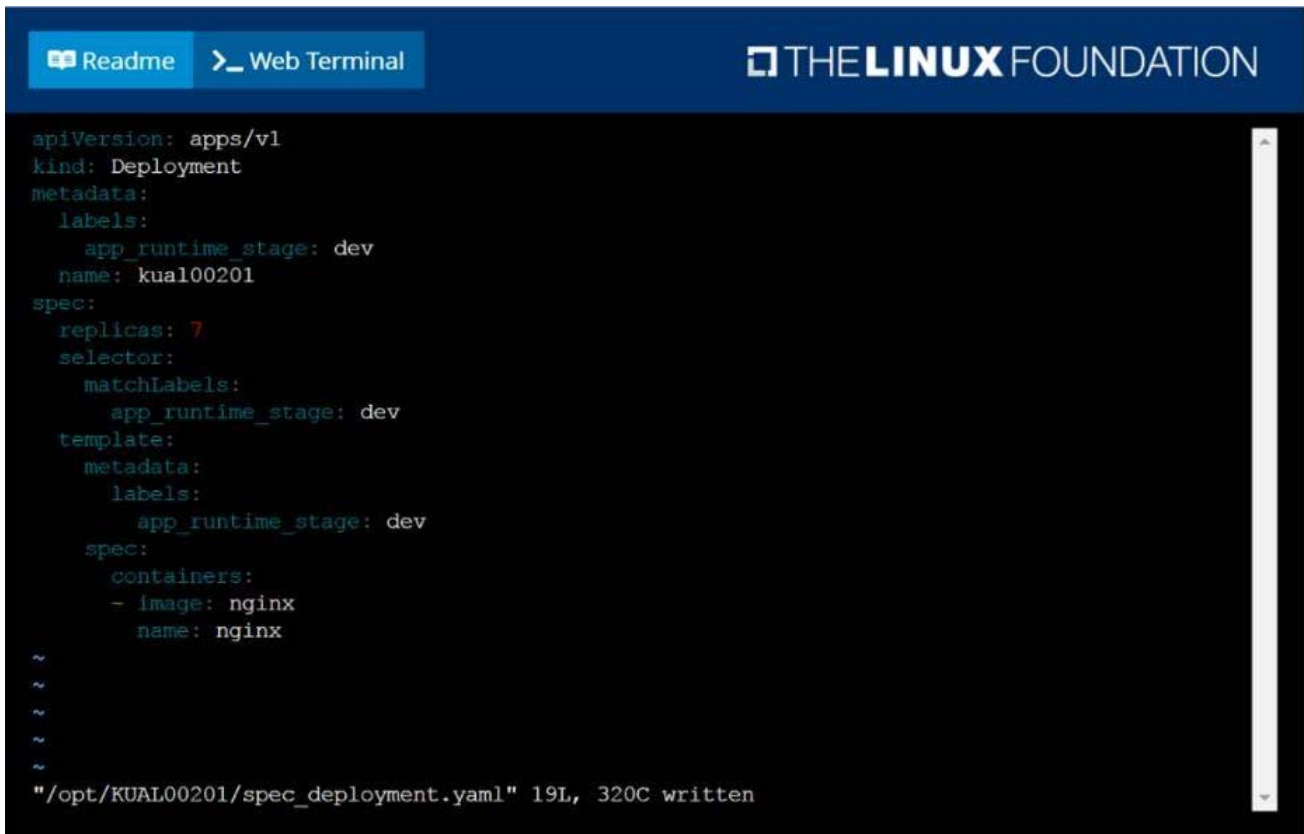
solution



The screenshot shows a web terminal window with a dark blue header. On the left, there are two buttons: 'Readme' and 'Web Terminal'. On the right, the logo for 'THE LINUX FOUNDATION' is displayed. The terminal content shows the following commands and output:

```
root@node-1:~# k create deploy kual00201 --image=nginx --dry-run=client -o yaml > /opt/KUAL00201/spec_deployment.yaml
root@node-1:~# vim /opt/KUAL00201/spec_deployment.yaml
```

The terminal cursor is positioned at the end of the second line, ready for input.



The screenshot shows a web terminal interface with a dark background and light-colored text. At the top, there are two buttons: "Readme" and "Web Terminal". To the right of these buttons is the logo for "THE LINUX FOUNDATION". The main area of the terminal displays a YAML configuration for a Kubernetes Deployment. The configuration includes fields for apiVersion, kind, metadata (with labels for app\_runtime\_stage and name), spec (with replicas, selector, and template), and a container definition for nginx. The terminal also shows a prompt character (~) and a message at the bottom indicating that the file was written successfully.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app_runtime_stage: dev
  name: kual00201
spec:
  replicas: 7
  selector:
    matchLabels:
      app_runtime_stage: dev
  template:
    metadata:
      labels:
        app_runtime_stage: dev
    spec:
      containers:
      - image: nginx
        name: nginx
~
~
~
~
~
~
"/opt/KUAL00201/spec_deployment.yaml" 19L, 320C written
```

### 3.CORRECT TEXT

From the pod label name=cpu-utilizer, find pods running high CPU workloads and write the name of the pod consuming most CPU to the file /opt/KUTR00102/KUTR00102.txt (which already exists).

**Answer:**

solution



⇒ Name: super-secret

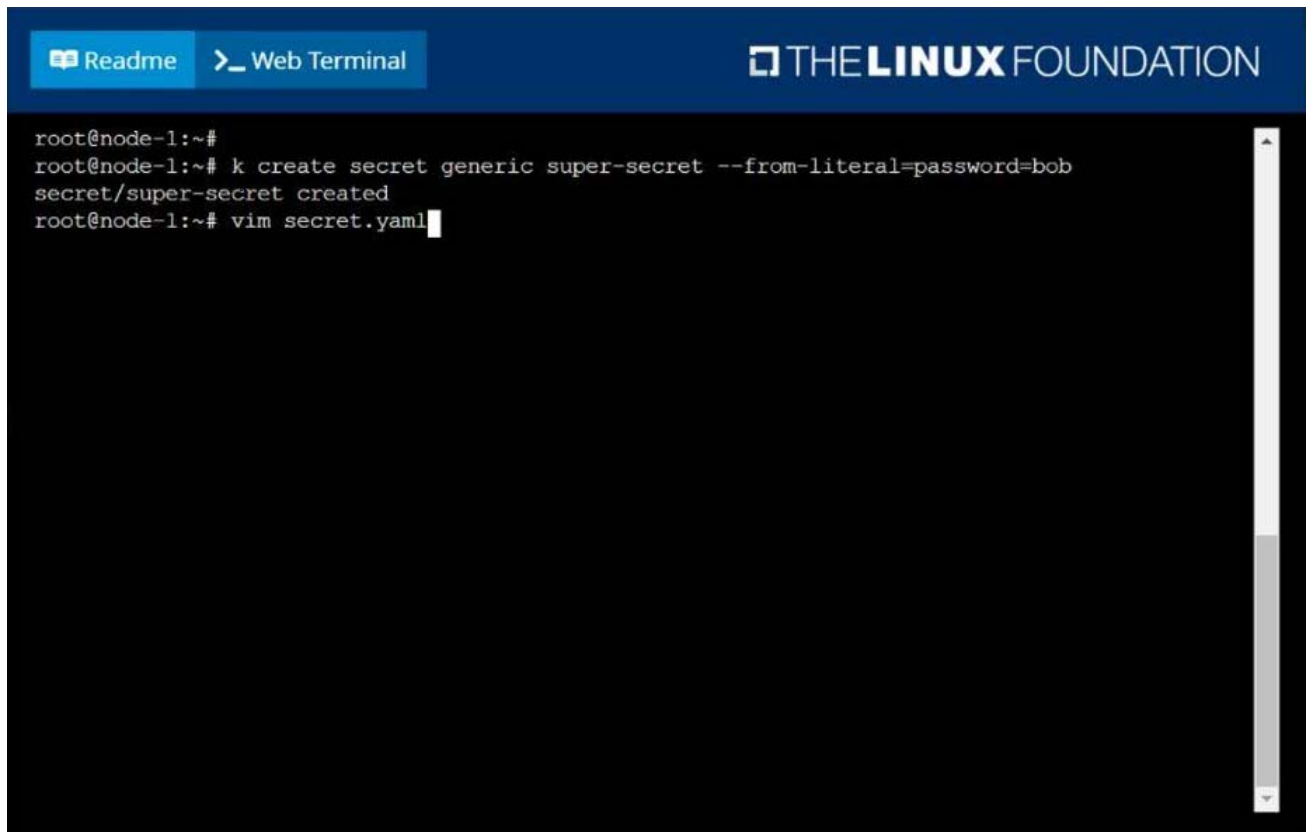
⇒ password: bob

Create a pod named pod-secrets-via-file, using the redis Image, which mounts a secret named super-secret at /secrets.

Create a second pod named pod-secrets-via-env, using the redis Image, which exports password as CONFIDENTIAL

**Answer:**

solution



The screenshot shows a web terminal interface with a dark blue header. On the left, there are two buttons: 'Readme' and 'Web Terminal'. On the right, the 'THE LINUX FOUNDATION' logo is displayed. The terminal content shows a series of commands and their outputs:

```
root@node-1:~#  
root@node-1:~# k create secret generic super-secret --from-literal=password=bob  
secret/super-secret created  
root@node-1:~# vim secret.yaml
```





```
Set configuration context:   
  
[student@node-1] $ | kube  
ctl config use-context k  
8s
```

**Task**

Create a new nginx Ingress resource as follows:

- Name: ping
- Namespace: ing-internal
- Exposing service hi on path /hi using service port 5678

```
The availability of service hi   
can be checked using the  
following command, which  
should return hi :  
  
[student@node-1] $ | curl  
-kL <INTERNAL_IP>/hi
```

**Answer:**

Solution:

```
vi ingress.yaml
```

```
#
```

```
apiVersion: networking.k8s.io/v1
```

```
kind: Ingress metadata: name: ping
```

```
namespace: ing-internal
```

```
spec:
```

```
rules:
```

```
- http:
```

```
paths:
```

```
- path: /hi pathType: Prefix backend: service: name: hi
```

```
port:
```

```
number: 5678
```

```
#
```

```
kubectl create -f ingress.yaml
```